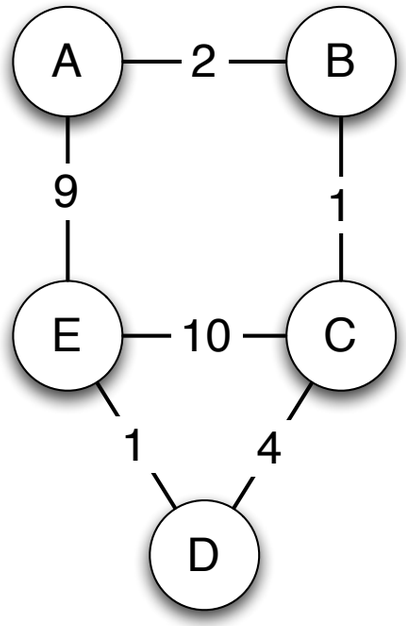
Distance Vector Routing Protocol

- Distance Table Datenstruktur
  - Jeder Knoten besitzt eine
    - Zeile für jedes mögliches Ziel
    - Spalte für jeden direkten Nachbarn
- Verteilter Algorithmus
  - Jeder Knoten kommuniziert nur mit seinem Nachbarn
- Asynchroner Betrieb
  - Knoten müssen nicht Informationen austauschen in einer Runde
- Selbst Terminierend
  - läuft bis die Knoten keine Informationen mehr austauschen



Distance Table für A

	über		Routing Tabellen
von A	B	E	Eintrag
nach B	**2**	15	B
C	**3**	14	B
D	**7**	10	B
E	**8**	9	E

Distance Table für C

	über			Routing Tabellen
von C	B	D	E	Eintrag
nach A	**3**	11	18	B
B	**1**	9	16	B
D	6	**4**	11	D
E	7	**5**	10	D

CoNe Freiburg

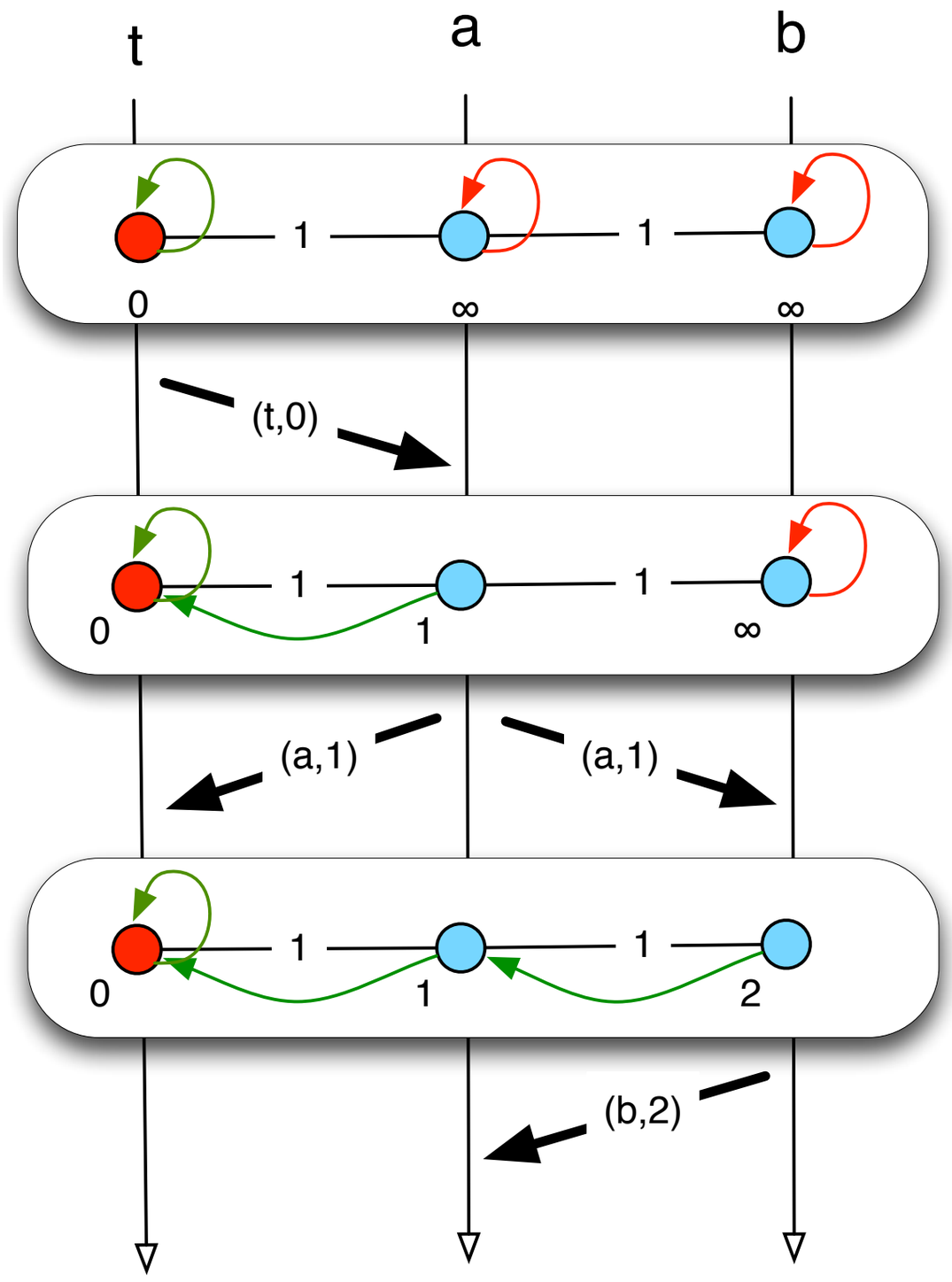
Distance Vector für ein Ziel t

Distributed Bellman Ford for target t (Distance-Vector Routing)

- If node is t then $d(t) \leftarrow 0$; $\pi(t) \leftarrow t$
- If a message from u to $\pi(u)$ fails then
  - $d(u) \leftarrow \infty$
- If u detects a new neighbor v then
  - send $(u, d(u))$ to v
- If u receives $(v, d(v))$ from v
  - if $d(u) > d(v) + w(u, v)$ or $v = \pi(u)$ then
    - $d(u) \leftarrow d(v) + w(u, v)$
    - $\pi(u) \leftarrow v$
- if $d(u) = \infty$ then $\pi(u) \leftarrow u$
- Every time $d(u)$ or $\pi(u)$ has changed u sends $(u, d(u))$ to all neighbors

CoNe Freiburg

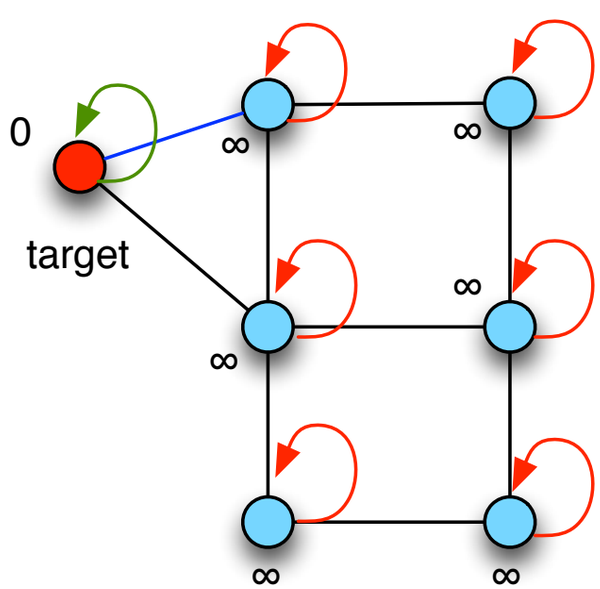
Beispiel für Distance-Vector für Ziel t

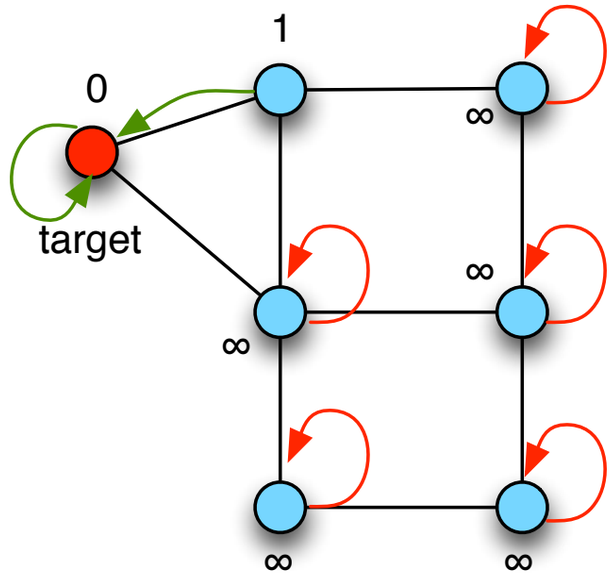


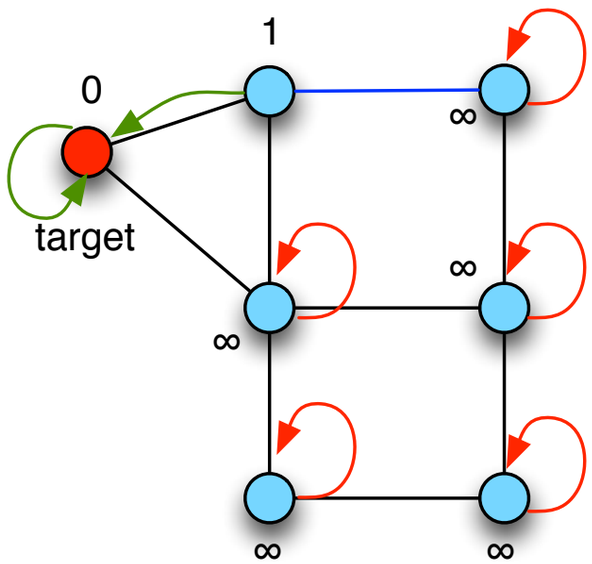
UNI FREIBURG

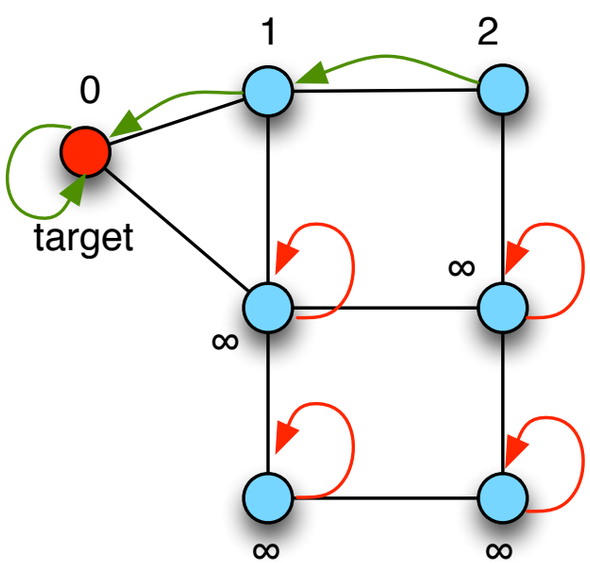
CoNe Freiburg

Distance-Vector für ein Ziel





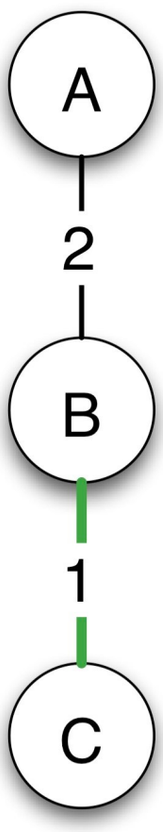




UNI FREIBURG

Das “Count to Infinity” - Problem

- Gute Nachrichten verbreiten sich schnell
  - Neue Verbindung wird schnell veröffentlicht



Distance Table für A

	über	
von A	B	Routing Tabellen Eintrag
nach B	**2**	B
C	**-**	B

Distance Table für B

	über		
von B	A	C	Routing Tabellen Eintrag
nach A	2	-	A
C	-	1	C

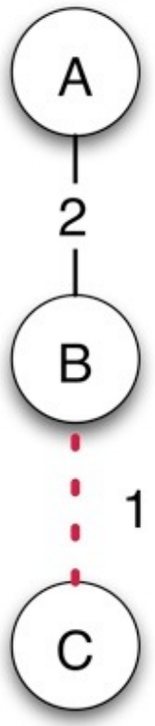
nach kurzer Zeit

	über	
von A	B	Routing Tabellen Eintrag
nach B	**2**	B
C	**3**	B

	über		
von B	A	C	Routing Tabellen Eintrag
nach A	2	4	A
C	5	1	C

Das “Count to Infinity” - Problem

- Schlechte Nachrichten verbreiten sich langsam
  - Verbindung fällt aus
  - Nachbarn erhöhen wechselseitig ihre Entfernung
  - “Count to Infinity”-Problem



von A	nach	über B	Routing Tabellen Eintrag
	B	2	B
	C	3	B

von B	nach	A	über C	Routing Tabellen Eintrag
	A	2	-	A
	C	5	-	A

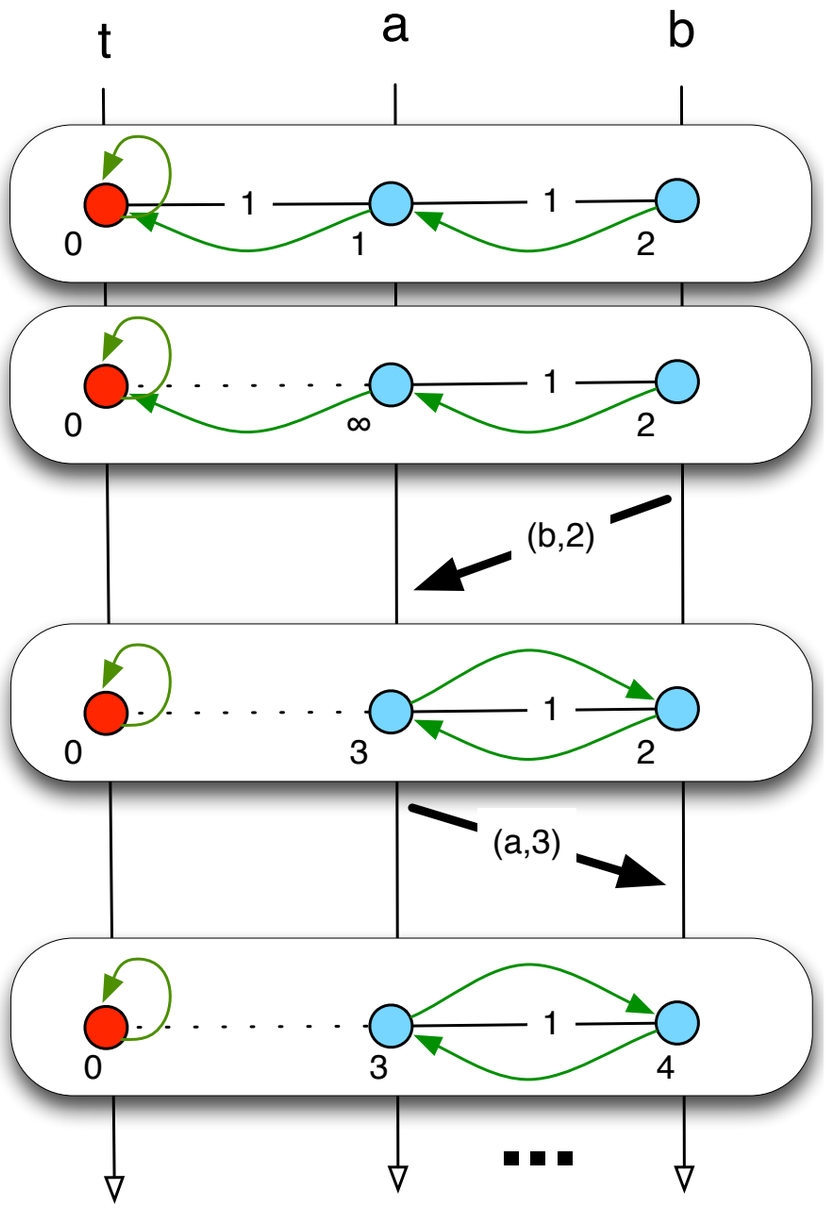
von A	nach	über B	Routing Tabellen Eintrag
	B	2	B
	C	7	B

von B	nach	A	über C	Routing Tabellen Eintrag
	A	2	-	A
	C	5	-	A

von A	nach	über B	Routing Tabellen Eintrag
	B	2	B
	C	7	B

von B	nach	A	über C	Routing Tabellen Eintrag
	A	2	-	A
	C	9	-	A

Das “Count to Infinity” - Problem für Ziel t



CoNe Freiburg

Link-State Protocol

- Link State Router
  - tauschen Information mittels Link State Packets (LSP) aus
  - Jeder verwendet einen eigenen Kürzeste-Wege-Algorithmus zu Anpassung der Routing-Tabelle
- LSP enthält
  - ID des LSP erzeugenden Knotens
  - Kosten dieses Knotens zu jedem direkten Nachbarn
  - Sequenznr. (SEQNO)
  - TTL-Feld für dieses Feld (time to live)
- Verlässliches Fluten (Reliable Flooding)
  - Die aktuellen LSP jedes Knoten werden gespeichert
  - Weiterleitung der LSP zu allen Nachbarn
    - bis auf den Knoten der diese ausgeliefert hat
  - Periodisches Erzeugen neuer LSPs
    - mit steigender SEQNOs
  - Verringern der TTL bei jedem Weiterleiten

UNI FREIBURG

CoNe Freiburg

Die Grenzen des flachen Routing

- Link State Routing
  - benötigt O(g n) Einträge für n Router mit maximalen Grad g
  - Jeder Knoten muss an jeden anderen seine Informationen senden
- Distance Vector
  - benötigt O(g n) Einträge
  - kann Schleifen einrichten
  - Konvergenzzeit steigt mit Netzwerkgröße
- Im Internet gibt es mehr als 10^7 Router
  - damit sind diese so genannten flachen Verfahren nicht einsetzbar
- Lösung:
  - Hierarchisches Routing